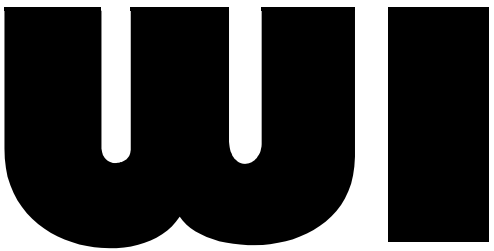
"Overview of the Document Object Model (DOM) a.k.a. DHTML Under Windows"

13th International Rexx Symposium, Raleigh, North Carolina
April 28th- May 1st, 2002

Rony G. Flatscher

Vienna University of Economics and Business Administration (WU)

Abteilung für Wirtschaftsinformatik



Agenda

- Terms
- DOM/DHTML
- Examples
- Roundup

Terms (Markup Languages) 1

- Tag
  - Allows to name markers which surround text
    - *Opening tag*
      - *<some_tag_name>*
    - *Closing tag*
      - *</some_tag_name>*
  - Allows to analyze text and determine which text portions are surrounded by which tags
- Element
  - The sequence "opening tag", text, "closing tag"

Terms (Markup Languages) 2

- Document Type Definition (DTD)
  - Defines Tags and their Attributes
    - *Names of tags*
      - *Defines attributes for tags*
    - *"Content model"*
      - *Nesting of tags and their sequence*
        - ***Hierarchical !***
      - *Determines the number of occurrences of tags*
  - "Instance" of a DTD
    - *A document with text that got marked up according to the DTD*
    - *A document which got checked according to the given DTD is called a "validated" document*

Terms (Markup Languages) 3

- HTML
  - A particular markup language for the WWW
    - *HTML-Browser*
      - *Parse HTML marked-up documents*
      - *Format text, depending on the tagged text*
  - DTD
    - *Version 4.01: three variants*
    - *SGML based, e.g.*
      - *Allows for case insensitive naming of tags and attributes*
      - *Some end-tags can be omitted, if the content model allows for inferencing them*
      - *Can define exclusion rules of tags*

Terms (Markup Languages) 4

- XML
  - A slightly simplified version of SGML
    - *Allows for defining DTDs for markup languages*
      - *Allows for defining XML Schemas as an alternative´*
    - *Names are case-sensitive*
    - *End tags must always supplied*
    - *Attribute values may be enclosed in apostrophes*
    - *Empty tags can be expressed*
      - *<some_tag_name/>*

Terms (Markup Languages) 5

- XML DTDs may be omitted
  - DTD can be inferred, if a document is "well-formed"
    - *Tags must be nested*
    - *Tags must not overlap*
    - *Opening tags must have matching end-tag*
- Structure separated from Formatting!
  - Cascading Style Sheets (CSS)
    - *Allow for defining formatting rules for tags*
    - *Possible to determine specific formatting rules based on attribute values or position of a tag within other tags!*

Example (HTML)

- A little text marked up with HTML

```
<html>
  <head>
     <title>This is my HTML file</title>
  </head>
  <body>
     <h1>Important Heading</h1>
     <p>This <span class="verb">is</span> the
        first paragraph.
     <h1>Another Important Heading</h1>
     <p id="xyz1">Another paragraph.
     <p id="9876">This <span class="verb">is</span> it.
  </body>
</html>
```

Example (HTML) With Cascading Style Sheet

- A little text marked up with HTML

```
<html>
  <head>
     <title>This is my HTML file</title>
     <link rel="stylesheet" type="text/css" href="example2.css">
  </head>
  <body>
     <h1>Important Heading</h1>
     <p>This <span class="verb">is</span> the
        first paragraph.
     <h1>Another Important Heading</h1>
     <p id="xyz1">Another paragraph.
     <p id="9876">This <span class="verb">is</span> it.
  </body>
</html>
```

Example (Cascading Style Sheet)

Tag

```
H1      { color: blue;
          text-align: center;
          font-family: Arial,sans-serif;
          font-size: 200%; }
```

Tag

```
body    { background-color: yellow;
          font-family: Times, Avantgarde;
          font-size: small; }
```

"class" attribute

```
.verb   { background-color: white;
          color: red;
          font-weight: 900; }
```

"id" attribute

```
#xyz1   { font-variant: small-caps;
          text-align: right; }
```

"id" attribute

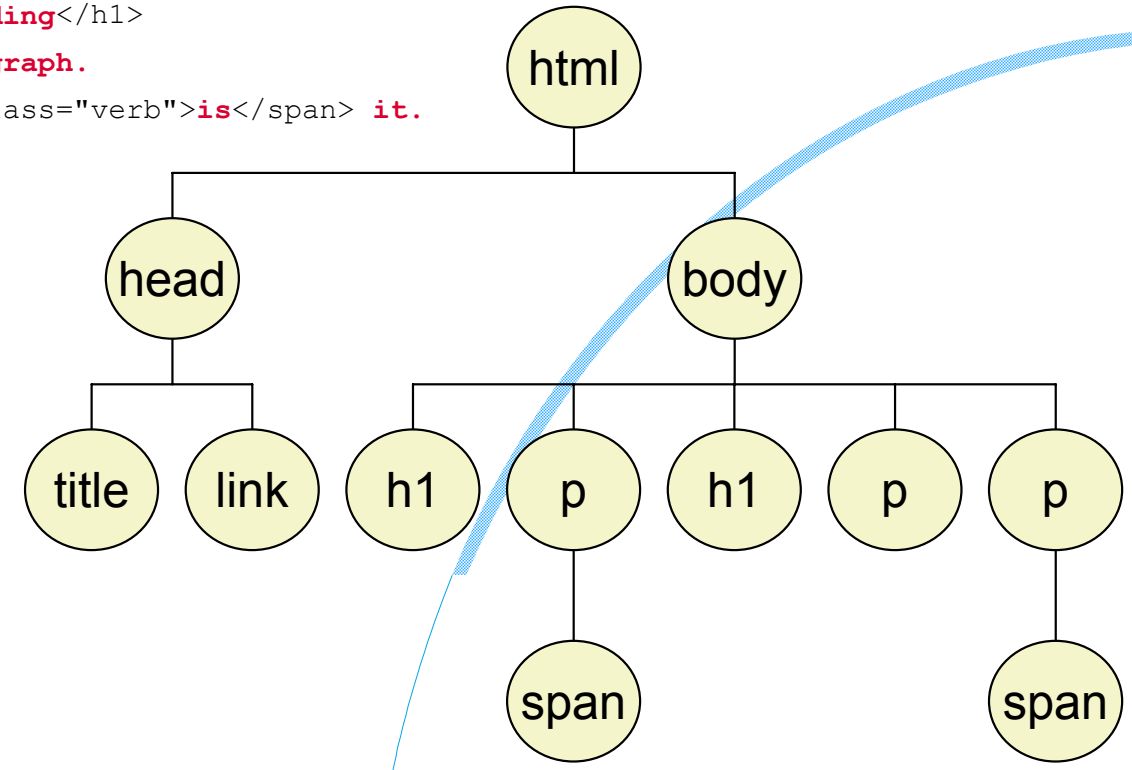
```
#9876   { font-size: large; }
```

Document Object Model (DOM)

- Parsing a HTML/XML file
  - Creates a parse tree with the elements as nodes
    - *Every HTML-browser has to do it with HTML files !*
- Application Programming Interface for
  - Building, Querying, Updating, Deleting the nodes in the parse tree
    - *Including the attributes of the tags !*
  - Intercepting events while working with the parse tree
    - *User generated events, like mouse-, key-actions*
    - *Application generated events like "document loaded"*

Document Object Model (DOM) Example

```
<html>
  <head>
     <title>This is my HTML file</title>
     <link rel="stylesheet" type="text/css" href="example2.css">
  </head>
  <body>
     <h1>Important Heading</h1>
     <p>This <span class="verb">is</span> the
        first paragraph.
     <h1>Another Important Heading</h1>
     <p id="xyz1">Another paragraph.
     <p id="9876">This <span class="verb">is</span> it.
  </body>
</html>
```



Adding Code to HTML/XML (1)

- <script>-Tag

```
<script language="Object Rexx"> -- <![CDATA[
      ... Rexx code ...
    -- ]]>

</script>
```

  or:

```
<script language="Object Rexx" src="file.rex"></script>
```

- "on..."-attributes

```
<some_tag onclick="call beep 90,90" language="Object Rexx">
```

  - One may call public Rexx routines
    - *"this"*
      - *Special variable referring to "this" element*
      - *May be used as an argument for the called routine*

Adding Code to HTML/XML (2)

- Microsoft's Internet Explorer
  - Windows Scripting Host
  - Incorporates the OLE-object "window"
    - *The Microsoft COM object for implementing DOM*
      - *"DHTML": dynamic HTML*
    - *Also all OLE-object properties of "window" are incorporated, e.g. "document"*
      - *Represents the HTML/XML document*
      - *Possesses functions/methods to e.g. get **all** nodes or all **tables** or elements of a certain type (tagname)*
      - *Allows for adding, updating, deleting elements*
  - Controls the execution of scripts
    - *Extraction, invocation*

Adding Code to HTML/XML (3)

- Code can be anywhere
  - Evaluated in document order
  - Public routines can be called from *any* other code
  - Code attached to event-attributes gets executed when the event fires

Example (HTML) With Rexx Code in It (1)

```
<html>
  <head>
     <script language="Object Rexx">
        document~writeln("<title>Produced by Rexx #1</title>")
     </script>
  </head>
  <body>
     <script language="Object Rexx">
        document~writeln("It is:<em>" date("s") time()"</em>, isn't it?")
     </script>
  </body>
</html>
```

Example (HTML) With Rexx Code in It (2)

```
<html>
  <head>
     <script language="Object Rexx">
        document~writeln("<title>Produced by Rexx # 2</title>")
        ::routine info public -- to be called later on
        use arg o
        window~alert("this=["o~tagName"] innerText=["o~innerText"]")
        window~alert("this=["o~tagName"] innerHtml=["o~innerHtml"]")
        window~alert("this=["o~tagName"] outerHtml=["o~outerHtml"]")
        tmpStr=""
        do item over document~all  -- iterate over all elements
           tmpStr=tmpStr item~tagName
        end
        call alert "elements:" tmpStr
     </script>
  </head>
  <body onclick="call info this" language="Object Rexx">
     <script language="Object Rexx">
        document~writeln("It is:<em>" date("s") time()"</em>, isn't it?")
     </script>
  </body>
</html>
```

Security Considerations

- MSIE
  - "Sandbox"
  - Object Rexx uses its "Security Manager" to adhere to
    - *e.g. accessing the Rexx environments may be forbidden*
    - *scripts are virtually relocated to the login dir*
    - *etc.*
- Local trusted execution only
  - Rename the "html" extension to "hta"
    - *"HTML Application"*

Roundup

- HTML/XML files
  - Hierarchical
  - Querying, adding, updating, removing elements from the parse tree
  - Adding code to events
- DOM
  - Document Object Model, W3C
  - DHTML
    - *Microsoft's implementation of DOM*
    - *Incomplete, proprietary extensions*
- Can be used for UI and printing !